Construction and ordering of edge elements for parallel computation

AUTHOR(S):

Iwashita, T; Shimasaki, M

Construction and Ordering of Edge Elements for Parallel Computation

Takeshi Iwashita, *Member, IEEE*, and Masaaki Shimasaki, *Member, IEEE*

***Abstract*—The present paper proposes a new method for the construction and ordering of edge elements for parallel computation. The use of virtual nodes generated in each volume element is presented as a means of introducing parallel ordering theory developed in finite difference analyses to finite edge element analyses. Eight-corner ordering and multi-color ordering are examined in the context of 3-D eddy-current analysis. The proposed method using 8-corner ordering can parallelize the ICCG solver in a finite edge element analysis without decreasing the convergence rate. A good balance between convergence and parallelism in the ICCG solver is obtained in the case of multi-color ordering.**

***Index Terms*—Edge element, parallelized ICCG method, parallel ordering, virtual node.**

I. Introduction

USE OF parallel orderings such as dissection ordering is an efficient way for parallelization of the ICCG method [1], [2]. It is, however, well-known that these ordering strategies entail a trade-off between parallelism and convergence. In order to overcome the trade-off, several detailed investigations have been performed in case of the finite difference method applied to elliptic partial differential problems [1], [3], [4]. No report, however, has been presented for finite element analyzes, in particular edge element analyses. The present paper investigates the effects of ordering strategies including domain division on parallelized finite edge element analyses. Since edge elements have complex data-dependency relationships to each other, the parallel ordering presented for the finite difference method cannot be directly applied to edge element analyses. On the other hand, edge elements are usually constructed in turn in each volume element such as a hexahedron. The present paper proposes a combination of the construction of edge elements and parallel ordering using virtual nodes generated in each volume element.

II. Parallel Processing of ICCG Method

The ICCG method is the most popular iterative solver for a symmetric positive-definite linear system arising in a FE analysis. The ICCG method involves four main kernels: 1) forward and backward substitutions, 2) inner products, 3) matrix–vector products, and 4) vector updates. The latter three kernels can be easily parallelized by dividing the vectors into several segments.

Manuscript received June 4, 2000.

T. Iwashita is with the Data Processing Center, Kyoto University, Kyoto 606-8501 Japan (e-mail: take@kudpc.kyoto-u.ac.jp).

M. Shimasaki is with the Department of Electrical Engineering, Kyoto University, Kyoto Japan (e-mail: simasaki@kuee.kyoto-u.ac.jp).

Publisher Item Identifier S 0018-9464(01)07854-2.

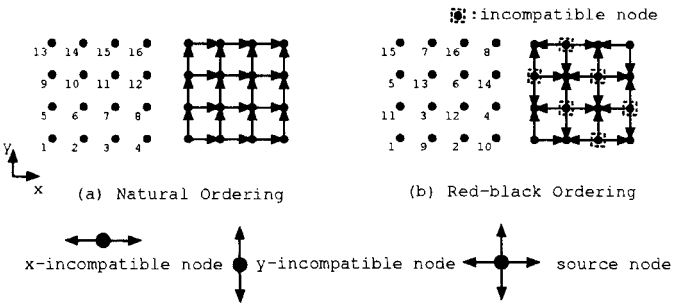

Fig. 1. Graph representation of ordering.

On the other hand, it is difficult to perform forward and backward substitutions in parallel.

The use of parallel ordering is one of the most effective methods for the parallel processing of substitutions. Parallel ordering, however, is well-known to entail a trade-off between parallelism and convergence. S. Doi and A. Lichnewsky give a good explanation of this trade-off in finite difference analyses by means of "the graph representation of ordering" [3]. Fig. 1 shows examples of the graph representation of ordering (natural ordering and red-black ordering). The directed graph represents the order between adjacent nodes; that is, the data-dependency relationship of the nodes. The node that is numbered before both sides of adjacent nodes in one direction is called "the incompatible node" (see Fig. 1). S. Doi and A. Lichnewsky have shown that the more incompatible nodes an ordering has, the lower the convergence rate is. Since natural ordering has no incompatible nodes, its convergence rate is excellent. Red-black ordering has many incompatible nodes, and its convergence rate is low. On the other hand, the number of incompatible nodes represents a potential parallelism. The incompatible node having no incoming edge in the ordering graph is called "the source node." The source nodes are potential starting points of the forward-backward substitution, and the number of source nodes shows ordering parallelism. This is the explanation of the trade-off between parallelism and convergence.

Two parallel ordering strategies are recommended from the view point of the theory of graph representation [4]. The first is 8-corner (4-corner in the 2-D case) ordering. This ordering has no incompatible node and the same degree of convergence rate as natural ordering. The number of processors is, however, limited. The second is multi-color ordering with more than 30 colors, though 2 or 4 colors are conventionally used [4]. This ordering is called "large-numbered multi-color ordering." Although this ordering suffers from the trade-off of the convergence rate, it attains a good balance between convergence and

parallelism. Accordingly, this ordering is effective in cases involving many processors (more than 8), in which 8-corner ordering is not applicable.

III. Ordering Strategy of Edge Elements for Parallel Processing

A. New Construction and Ordering of Edge Elements

The present paper investigates an application of the parallel ordering theory developed for finite difference analyses, as mentioned above, to finite edge element analyses. Since edge elements have complex data-dependency relationships each other, it is difficult to directly apply parallel ordering to edge elements. We, here, propose a new construction and ordering procedure of edge elements for parallel processing using virtual nodes generated in volume elements.

Edge elements are usually constructed in each volume element. In the proposed method, virtual nodes are, first, generated at the center of gravity in volume elements, and then, parallel ordering is applied to the virtual nodes. Edge elements are constructed following the ordering of virtual nodes. The procedure is summarized as follows:

Step 1) A virtual node is generated at the center of gravity in each volume element.
Step 2) The virtual nodes are ordered corresponding to a parallel ordering strategy.
Step 3) Edge elements are constructed in each volume element following the ordering of the virtual nodes.

Eight-corner ordering and large-numbered multi-color ordering are used in the present analyses. The details of the construction procedure using these orderings are described in the following subsections.

B. Construction of Edge Elements Using 8-Corner Ordering

For the sake of simplicity, we illustrate the procedure in the 2-D case while the present analysis is 3-D. Fig. 2(a) shows an example of the analytical models. We here use the 4-corner (8-corner in 3-D) ordering in Step 2) mentioned above. The virtual nodes are divided into four subdomains each of which includes a corner node as is shown in Fig. 2(b). A subdomain is assigned to each processor. In each subdomain, the nodes are ordered using natural ordering where the corner node is selected as the starting point. Since the ordering graph of the virtual nodes has no incompatible nodes, the convergence rate is expected to be excellent. After the construction of edge elements in Step 3), we find the interface elements that belong to several different subdomains. The interface elements are excluded from the subdomains and are renumbered next to the remaining interior elements of the subdomains. Since this procedure leads to a global matrix similar to a dissection ordering case, the forward-backward substitution can be performed in parallel. If required, the interface elements are also subdivided into several groups.

In the present 3-D analysis, most of the interface elements belonging to two subdomains are divided into different groups sharing no data-dependency with each other. The forward–backward substitution concerned with these elements is carried out in parallel. The remaining interface elements, such as those shared by more than three subdomains, are treated sequentially by all processors.

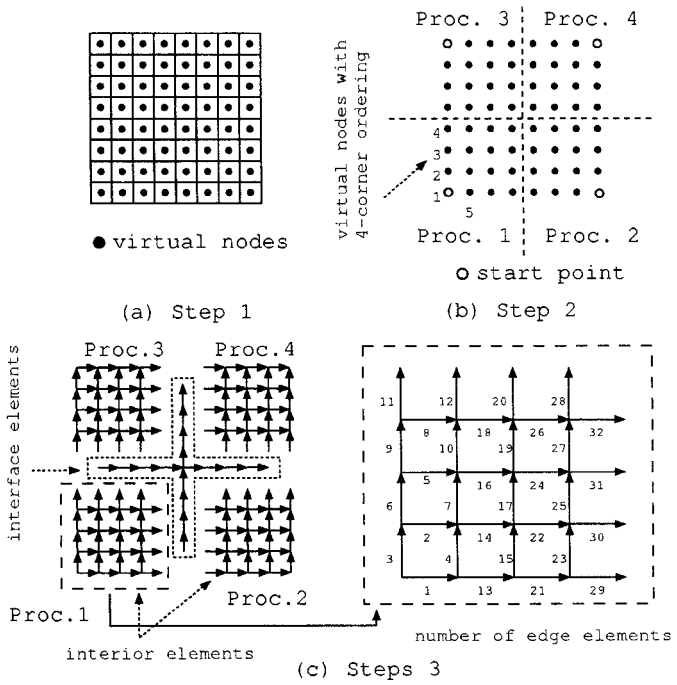

Fig. 2. Construction and ordering using 4-corner ordering.

C. Construction of Edge Elements With Multi-Color Ordering

In this subsection, we discuss the case of multi-color(m) ordering where m denotes the number of colors. In Step 2), the virtual nodes are ordered first using natural ordering or diagonal ordering. The present analysis adopts diagonal ordering. Second, several colors are assigned to the virtual nodes, where the virtual nodes with the same color must be independent. This means that edge elements generated in the volume elements of the same color must not share data-dependency each other. In Step 3), edge elements are constructed in volume elements following the diagonal ordering of virtual nodes. When the edge element is newly constructed, the same color as the virtual node of the volume element is assigned to the edge element. Next, edge elements are reordered following the order of color from $C(1)$ to $C(m)$, where $C(i)$, $(i = 1, 2, \ldots, m)$ represents the set of edge elements sharing the same color i. The edge elements constructed in different volume elements of the same color can be treated in parallel. Consequently, the degree of the parallelism is equal to the number of virtual nodes assigned in one color.

Fig. 3 illustrates the procedure using multi-color(3) ordering for a 2-D rectangular mesh as in Fig. 2(a). Fig. 3(a) shows the virtual nodes using diagonal ordering. Fig. 3(b) depicts the color of the virtual node. Fig. 3(c) and (d) show the color and the final order of edge elements constructed in volume elements, respectively. Fig. 4 depicts the global matrix based on the final ordering of edge elements.

In the case of multi-color ordering, the number of colors has a lower boundary in order to attain the independence of virtual nodes with the same color. The lower limit value depends on the virtual node ordering and the type of the volume element. About 30 colors are sufficient for hexahedron and tetrahedron

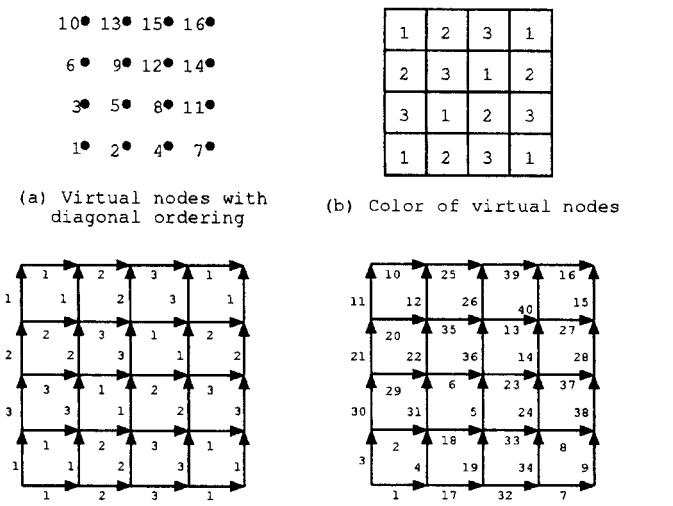

Fig. 3. Construction and ordering using multi-color(3) ordering.

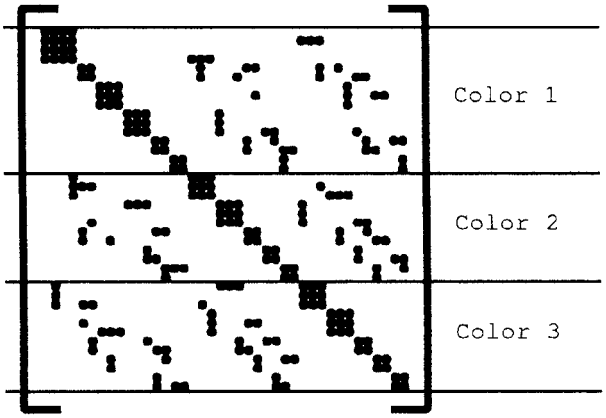

Fig. 4. Global matrix with multi-color(3) ordering.

elements while from 30 to 50 colors are usually used in the large-numbered multi-color ordering. The use of too many colors is, however, ineffective because multi-color(m) ordering requires $2(m-1)$-times communications in one set of forward and backward substitutions.

IV. Parallel Performance

A. Test Model and Computation Environment

We have implemented the proposed method for the IEEJ standard benchmark model of 3-D eddy current analyses [5]. Fig. 5 shows the analyzed model, which is discretized by first-order brick-type edge elements in the present analysis. Table I lists the discretization data. The electromagnetic field equations are formulated with A-formulation, and are solved by the Galerkin method and the backward time difference method.

Parallelized eddy current analyses were carried out on the distributed memory parallel computer Fujitsu VPP-800 at the Data Processing Center, Kyoto University. The program code is written in the FORTRAN language with the MPI library. The convergence criterion of the ICCG method is given by $||\mathbf{r}||_2/||\mathbf{b}||_2 < 10^{-7}$, where $\mathbf{b}$ and $\mathbf{r}$ are the right-hand side vector and the residual vector, respectively. The acceleration

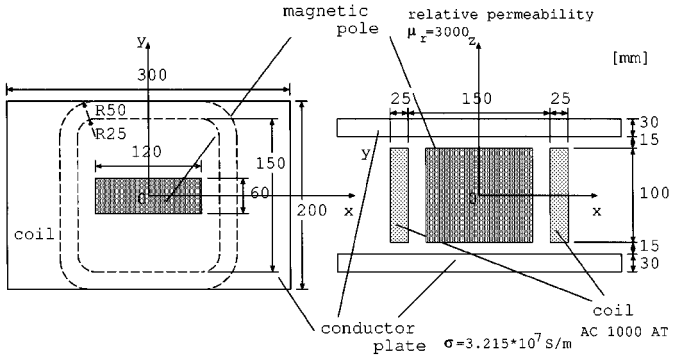

Fig. 5. IEEJ standard benchmark model (unit: [mm]).

TABLE I
Analysis Conditions and Discretization Data

number of volume elements	10600
number of nodes	12100
number of unknowns	34650
time step	1 msec

TABLE II
Computation Results

(a) Proposed method using n-corner ordering

N_p	Computation time (sec)	Iteration
1	13.46	72
2	6.979	72
4	3.411	71
8	2.035	73

(b) Natural ordering without virtual nodes

N_p	Computation time (sec)	Iteration
1	15.89	85

factor of the ICCG method is chosen to be 1.03. The parallel performance of the proposed method is evaluated in the first one time step.

B. 8-Corner Ordering

This subsection examines the parallel performance of the proposed method when the virtual nodes are ordered with n-corner ordering (n processors), $n = 8, 4, 2, 1$. One-corner ordering here means natural ordering. Table II lists the computation time and CG iterations in the first time step, where N_p is the number of processors. For comparison, Table II(b) shows the computation results without virtual nodes on one processor where each edge element is ordered using natural ordering. Table II shows that the finite edge element analysis can be parallelized without decreasing the convergence rate. Table II also implies that the use of virtual nodes is effective even in the one processor case, because more CG iterations are required in the case of no virtual node. Fig. 6 depicts the convergence behavior showing that the n-corner ordering preserves a good convergence rate. Fig. 7 shows the speed-up ratio compared with the case of no virtual node. The proposed method achieves high parallel performance because of no decline of the convergence rate. The saturation

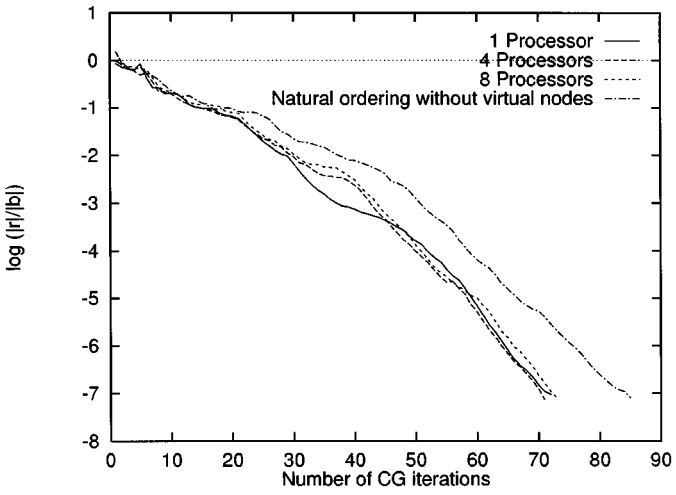

Fig. 6. Convergence behavior with n-corner ordering.

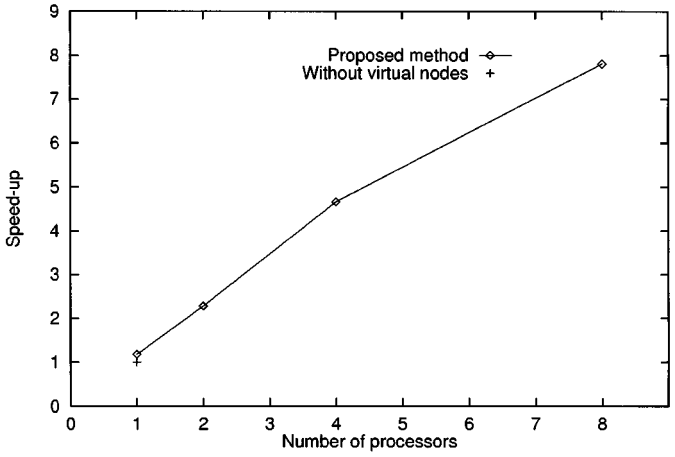

Fig. 7. Speed-up of proposed method using n-corner ordering.

TABLE III
DISCRETIZATION DATA OF LARGE-SCALE PROBLEM

number of volume elements	327680
number of nodes	342225
number of unknowns	1011920

of the speed-up on 8 processors is caused by an increase in the ratio of communication cost to computation cost. Higher parallel performance is expected for a larger-scale problem.

C. Multi-Color Ordering

Since large-numbered multi-color ordering is effective for a large-scale problem, the analyzed model discretized by a fine mesh is presented for a large-scale model in this subsection. The discretization data are listed in Table III. The acceleration factor of the ICCG method is set at 1.07.

When the virtual node is ordered with multi-color ordering, the convergence of the ICCG solver depends not on the number of processors but on the number of colors. Fig. 8 shows the convergence behavior of the proposed method using several numbers of colors. In testing various numbers of colors from

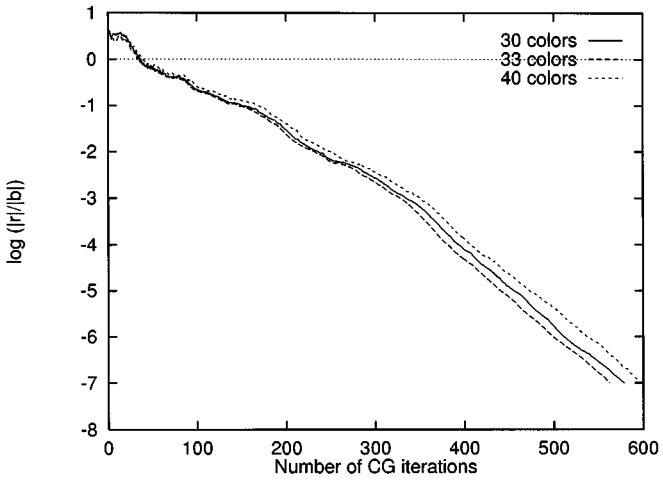

Fig. 8. Convergence behavior with multi-color ordering.

TABLE IV
COMPUTATION RESULTS FOR LARGE-SCALE PROBLEM

(a) Proposed method using multi-color(33) ordering

N_p	Computation time (sec)	Iteration
1	3340	563
4	857	563
8	436	562
16	227	562
24	161	563
32	130	563

(b) Natural ordering without virtual nodes

N_p	Computation time (sec)	Iteration
1	2072	366

(c) Proposed method using n-corner ordering

N_p	Computation time (sec)	Iteration
1	1875	327
4	512	349
8	257	337

27 to 50, we obtained the highest convergence rate and parallel performance when 33 colors were used. Table IV lists the computation time and CG iterations in the first time step. Table IV indicates that the proposed method using multi-color(33) ordering suffers from about a 50% increase of CG iterations compared with the natural ordering case on one processor. On the other hand, in the case of 33 colors, about 9900 virtual nodes are assigned to one color; that is, the ordering attains about 9900 degrees of parallelism. Large-numbered multi-color ordering can achieve high parallelism with a small decrease of convergence rate.

We have also examined the application of n-corner ordering for this large-scale problem. Fig. 9 plots the speed-up ratio of the proposed method compared with the case of no virtual nodes. It is shown that n-corner ordering achieves an ideal speed-up for the large-scale problem. Multi-color ordering is effective on more than 8 processors, and obtains a 16.0-fold speed-up by 32 processors.

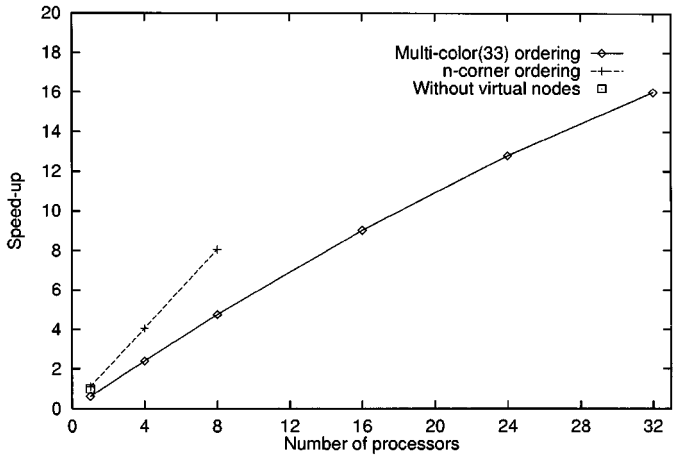

Fig. 9. Comparison of speed-up in large-scale problem.

V. Conclusion

The present paper proposes the use of virtual nodes in the application of parallel ordering techniques to edge element analyses. The proposed method is evaluated in terms of 3-D eddy current analyses, demonstrating that the useful properties which parallel ordering techniques exhibit in finite difference analyzes can be effectively introduced to finite edge element analyzes. Using 8-corner ordering, an edge element analysis is parallelized with no decline of the convergence rate of the ICCG solver. When the multi-color ordering is applied to the virtual nodes, a good balance between convergence and parallelism is attained in the solver.

References

[1] I. S. Duff and G. A. Meurant, "The effect of ordering on preconditioned conjugate gradients," *BIT*, vol. 29, pp. 635–657, 1989.
[2] K. Iwano, V. Cingoski, K. Kaneda, and H. Yamashita, "A parallel processing method in finite element analysis using domain division," *IEEE Trans. Magn.*, vol. 30, no. 5, pp. 3598–3601, Sept. 1994.
[3] S. Doi and A. Lichnewsky, "A graph-theory approach for analyzing the effects of ordering on ILU preconditioning," INRIA, report 1452, 1991.
[4] S. Doi and T. Washio, "Ordering strategies and related techniques to overcome the trade-off between parallelism and convergence in incomplete factorization," *Parallel Computing*, vol. 25, pp. 1995–2014, 1999.
[5] T. Nakata, N. Takahashi, T. Imai, and K. Muramatsu, "Comparison of various methods of analysis and finite elements in 3-D magnetic field analysis," *IEEE Trans. Magn.*, vol. 27, no. 5, pp. 4073–4076, Sept. 1991.
[6] R. Barrett, *et al.*, "Templates for the solution of linear systems: Building blocks for iterative methods," *SIAM*, 1994.